

PATCH DIFFING 101

ABOUT ME – KOREY YOUNG

- Bloomsburg University – Computer Science.
- But I have always had an interest in cyber security and hacking / red team activities
 - Finding ways to make computers do things that they were not meant to do;
 - So that we can get it fixed before the bad guys discover it and try to use it.
- Dakota State University - Masters in Information Assurance
- SANS GWAPT, Sec 642, GPEN, GXPN, Sec 760
- My day job is penetration testing for an IT company, trying to hack into our applications.
- In my free time after work and on weekends, I do cyber security and exploit research.

PATCH DIFFING

- How do reverse engineers find what changed in patched software, in order to exploit it?
- They use Patch Differencing, which is when you use a tool to examine the contents of 2 binary files, to see what changed between them

PATCH DIFFING

- Hex Rays' BinDiff is one of the most well known tools to use to do this
 - Also Radiff2 from Radare and Diaphora
- You can manually import the resulting diff files into a decompiler such as IDAPro or Ghidra
 - Or IDAPro professional does the diffing via a menu option, via a plugin with BinDiff

PATCH DIFFING

- Red = more code changes
- Green = less code changes

ida64 File Edit Jump Search View Debugger Lumina Options Windows Help

IDA - ntoskrnl_2022-01.exe /Users/kyoung/Desktop/securityCourses/sec760/labFiles/Files_On_Windows_VM/day4/ntoskrnl_2022-01.exe

No debugger

Library function Regular function Instruction Data Unexplored External symbol Lumina function

Fun IDA ... Hex ... Stru... Enums Imp... > Matched Function... Statisti... Primary Unmatched Secondary Unmatch...

Function name

- _RtlpMuiRegInit!
- _RtlpMuiRegPop!
- _RtlpMuiRegVali
- _RtlpMuiRegVali
- EtwpEventTracir
- LkmdTelCreateR
- LkmdTelSubmitF
- LkmdTelpWriteD
- nullsub_20
- nullsub_21
- nullsub_22
- nullsub_23
- nullsub_24
- nullsub_25
- sub_140987050

Line 21896 of 24847

Gra Output

100.00% (-377,147) 008DF210 000000014098C010: KiS (Synchronized with Hex)

INSTRUCTIONS: Discarding.

Function 000000014098C010 is excessively large: 2631 basic blocks, 4133 edges, 11812 instructions. Discarding.

6.1s for matching.

Caching 'Matched Functions'... ok

Python

AU: idle Down Disk: 295GB

Similarity ^ | Confide | Change | EA Primary | Name Primary | EA Secondary

Similarity ^	Confide	Change	EA Primary	Name Primary	EA Secondary
0.99	0.99	GI-J---	00000001406B...	PspBuildCreateProcessContext	000000014...
0.98	0.99	GI----	000000014061...	NtCreateUserProcess	000000014...
0.98	0.99	GI----	000000014098...	PopTransitionSystemPowerStateEx	000000014...
0.98	0.99	-I-J---	000000014079...	ExInitializeLeapSecondData	000000014...
0.98	0.99	GI----	0000000140A3...	Phase1InitializationDiscard	000000014...
0.98	0.99	GI-E--	000000014061...	PspAllocateProcess	000000014...
0.97	0.97	-I-JE--	000000014099...	ExpSetSystemTime	000000014...
0.94	0.99	GI----	000000014079...	ExpReadLeapSecondData	000000014...
0.94	0.98	GI-JE--	000000014061...	SeSubProcessToken	000000014...
0.92	0.93	-I-JE-C	000000014098...	ExUpdateSystemTimeFromCmos	000000014...
0.89	0.96	GI-E--	000000014099...	EmonRestartProfiling	000000014...
0.85	0.95	GI-JE--	00000001406E...	IopGraftName	000000014...
0.85	0.98	GI-E--	000000014079...	ExpRefreshTimeZoneInformation	000000014...
0.77	0.97	GI-E--	000000014038...	EmonConfigureCounter	000000014...
0.72	0.94	GI-JE--	0000000140A3...	PolnitSystem	000000014...
0.45	0.99	G-----	000000014032...	WarbirdCrypto__CCipherFeistel64__CallRoundFunc...	000000014...
0.21	0.73	GI-EL--	000000014091F...	SepAddTokenOriginClaim	000000014...
0.04	0.06	GI-E--	00000001403C...	PopCheckForAbnormalReset	000000014...
1.00	0.99	-----	0000000140131...	_imp_VidInitialize	000000014...
1.00	0.99	-----	0000000140131...	_imp_VidBufferToScreenBit	000000014...
1.00	0.99	-----	0000000140131...	_imp_VidScreenToBufferBit	000000014...
1.00	0.99	-----	0000000140131...	_imp_VidBitBlit	000000014...
1.00	0.99	-----	0000000140131...	_imp_VidResetDisplay	000000014...
1.00	0.99	-----	0000000140131...	_imp_VidCleanUp	000000014...
1.00	0.99	-----	0000000140131...	_imp_VidSetTextColor	000000014...
1.00	0.99	-----	0000000140131...	_imp_VidSetScrollRegion	000000014...
1.00	0.99	-----	0000000140131...	_imp_VidSolidColorFill	000000014...
1.00	0.99	-----	0000000140131...	_imp_VidDisplayString	000000014...
1.00	0.99	-----	0000000140131...	_imp_VidBitBltEx	000000014...
1.00	0.99	-----	0000000140131...	_imp_Clinitialize	000000014...
1.00	0.99	-----	0000000140131...	_imp_PshedDoPfa	000000014...
1.00	0.99	-----	0000000140131...	_imp_PshedGetErrorSourceInfo	000000014...

Line 1 of 26555



PATCH DIFFING

- Primary Unmatched = functions only found in the first of the diffed files
- Secondary Unmatched = functions only found in the second of the diffed files

File Edit Jump Search View Debugger Lumina Options Windows Help

ntoskrnl_2022-01.exe /Users/kyoung/Desktop/securityCourses/sec760/labFiles/Files_On_Windows_VM/day4/ntoskrnl_2022-01.exe

No debugger

Library function Regular function Instruction Data Unexplored External symbol Lumina function

Fun IDA ... Hex ... Stru... Enums Imp... Matched Function... Statistic... Primary Unmatched Secondary Unmatched

Function name

- _RtlpMuiRegInitl
- _RtlpMuiRegPop
- _RtlpMuiRegVali
- _RtlpMuiRegVali
- EtwpEventTracir
- LkmdTelCreateR
- LkmdTelSubmitf
- LkmdTelpWriteD
- nullsub_20
- nullsub_21
- nullsub_22
- nullsub_23
- nullsub_24
- nullsub_25
- sub_140987050

Line 21896 of 24847

Gra

100.00% (-377,147) 008DF210 000000014098C010: KiS (Synchronized with He

Output

INSTRUCTIONS: Discarding.

Function 00000001403DAA30 is excessively large: 2631 basic blocks, 4133 edges, 11812 instructions. Discarding.

6.1s for matching.

Caching 'Matched Functions'... ok

Python

AU: idle Down Disk: 295GB

Icons: Finder, App Store, Settings, Safari, Google Chrome, Firefox, VS Code, Notes, Reminders, Stocks, iBooks, Mail, Calendar, Photos, Files, Archive, Trash

EA	Name	Basic Block	Instructions	Edges
00000001403A...	HvllsPartitionCpuManager	1	4	0
00000001403E...	Feature_Servicing_CorrectTimeAfterDST_3380945...	1	24	0
00000001404E...	HvllsAnyHypervisorPresent	1	5	0
0000000140A6...	PopCheckForUpgradeInProgress	7	68	11
0000000140A6...	PpmCheckInit	1	15	0
0000000140A7...	PpmWmiInit	1	16	0
0000000140A7...	PopCoolingInit	1	6	0
0000000140A7...	PopTransitionCheckpointInit	1	6	0
0000000140A7...	PpmEventInitialize	3	12	3
0000000140A7...	PopReadErrataDisablePrimaryDeviceFastResume	3	11	3
0000000140A7...	PopDripsWatchdogInitialize	7	20	10
0000000140A7...	PopRecordFirmwareResetReason	3	13	2
0000000140A7...	PopCheckpointDetermineEfiRuntimeRedirection	1	5	0
0000000140A7...	PopInitModernSleepEnabledActions	3	5	3
0000000140A7...	PopReadErrataForIncorrectLidNotification	4	13	5
0000000140A7...	PopEvaluateInputSuppressionRequired	7	32	10
0000000140A9...	PopCoolingInitializeWnfEvents	1	5	0
0000000140A9...	PopInitializePdc	1	3	0
0000000140A9...	PopPolicyDeviceInit	1	3	0
0000000140A9...	PopFanInit	1	3	0
0000000140A9...	PopEmRegister	1	11	0

ida64 File Edit Jump Search View Debugger Lumina Options Windows Help

IDA - ntoskrnl_2022-01.exe /Users/kyoung/Desktop/securityCourses/sec760/labFiles/Files_On_Windows_VM/day4/ntoskrnl_2022-01.exe

No debugger

Library function Regular function Instruction Data Unexplored External symbol Lumina function

Fun IDA ... Hex ... Stru... Enums Imp... Matched Function... Statistic... Primary Unmatched Secondary Unmatched

Function name

- _RtlpMuiRegInit!
- _RtlpMuiRegPop!
- _RtlpMuiRegVali
- _RtlpMuiRegVali
- EtwpEventTracir
- LkmdTelCreateR
- LkmdTelSubmitF
- LkmdTelWriteD
- nullsub_20
- nullsub_21
- nullsub_22
- nullsub_23
- nullsub_24
- nullsub_25
- sub_140987050

Line 21896 of 24847

Gra

Output

INSTRUCTIONS: Discarding.

Function 00000001403DAA30 is excessively large: 2631 basic blocks, 4133 edges, 11812 instructions. Discarding.

6.1s for matching.

Caching 'Matched Functions'... ok

AU: idle Down Disk: 295GB

100.00% (-377,147) 008DF210 000000014098C010: KiS (Synchronized with He)

Code view:

```
; NSTATUS __stdcall KiSystemStartup(PDRIVER_OBJECT)
public KiSystemStartup
KiSystemStartup proc near
var_1C8= qword ptr -1C8h
var_1C0= qword ptr -1C0h
var_8= qword ptr -8

sub    rsp, 38h
mov    [rsp+38h+var_8], r15
mov    r15, rsp
mov    cs:KeLoaderBlock_0, rcx
mov    ecx, 0FFFFFFFh
mov    rdx, cs:KeLoaderBlock_0
mov    r10, [rdx+88h]
cmp    dword ptr [r10+24h], 0
jnz    short loc_14098C042

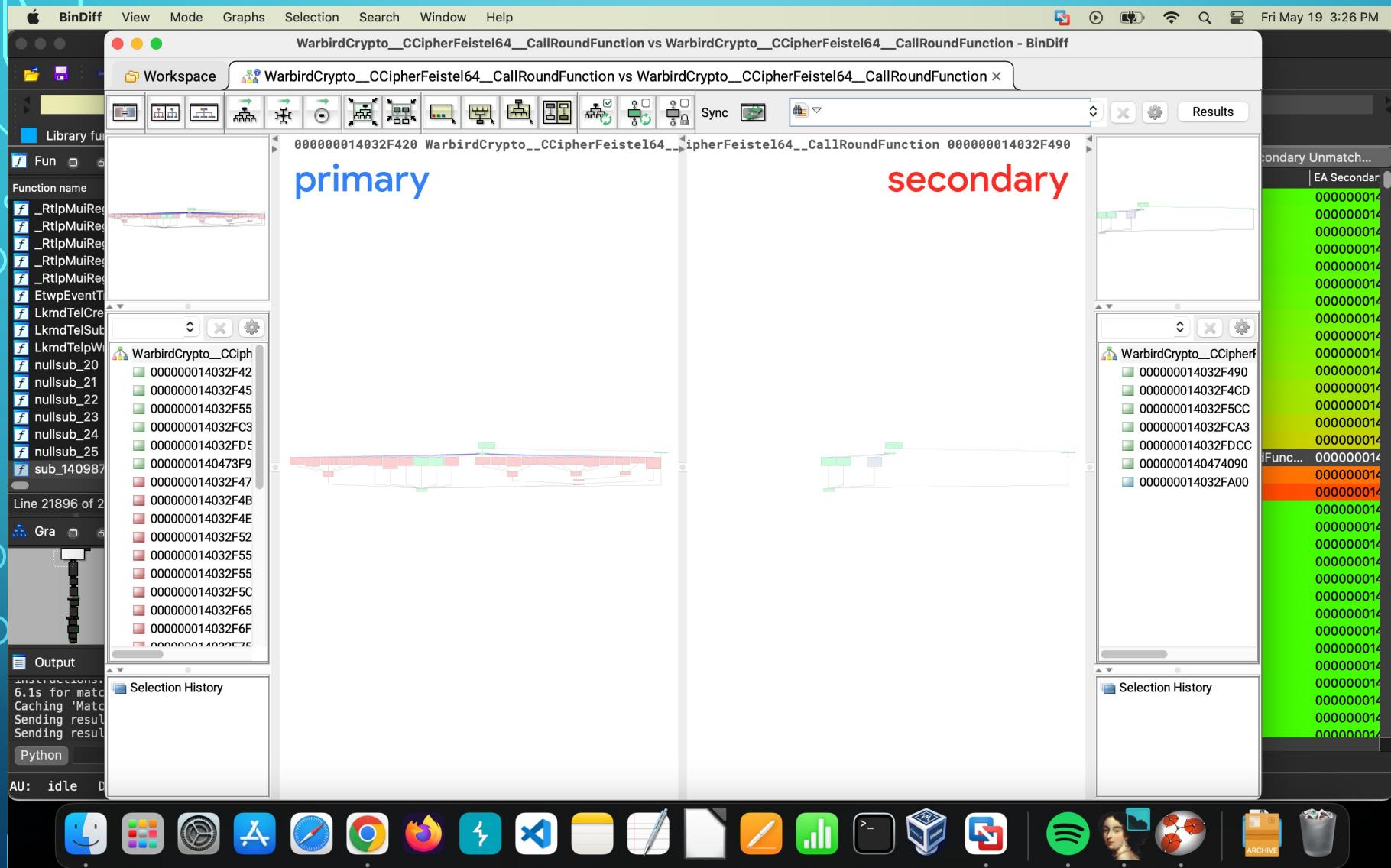
loc_14098C042:
mov    rcx, cs:KeLoaderBlock_0
mov    rdx, [rcx+88h]
mov    r10, rdx
sub    rdx, 180h
mov    [rdx+18h], rdx
mov    [rdx+20h], r10
mov    r8, cr0
mov    [rdx+28h], r8
mov    r8, cr2
mov    [rdx+28h], r8
```

Table view:

EA	Name	Basic Block	Instructions	Edges
00000001405C...	SeDeleteCodeIntegrityOriginClaimMembers	3	7	3
00000001405C...	SeGetCodeIntegrityOriginClaimForFileObject	4	9	4

PATCH DIFFING

- You can drill down into the diff report, and view the added/removed/modified code sections
 - Green = added code sections
 - Red = removed code sections
 - Yellow = modified code sections



BinDiff View Mode Graphs Selection Search Window Help

Fri May 19 3:24 PM

PopCheckForAbnormalReset vs SeDeleteCodeIntegrityOriginClaimForFileObject - BinDiff

Workspace PopCheckForAbnormalReset vs SeDeleteCodeIntegrityOriginClaimForFileObject

Sync Results

secondary Unmatch... EA Secondary

00000001403CC964 PopCheckForAbnormalReset IntegrityOriginClaimForFileObject 00000001405C8918

primary secondary

Function name

- _RtlpMuiReq
- _RtlpMuiReq
- _RtlpMuiReq
- _RtlpMuiReq
- _RtlpMuiReq
- EtwpEventT
- LkmdTelCre
- LkmdTelSub
- LkmdTelWP
- nullsub_20
- nullsub_21
- nullsub_22
- nullsub_23
- nullsub_24
- nullsub_25
- sub_140987

Line 21896 of 2

Gra Output Selection History

PopCheckForAbnormalR

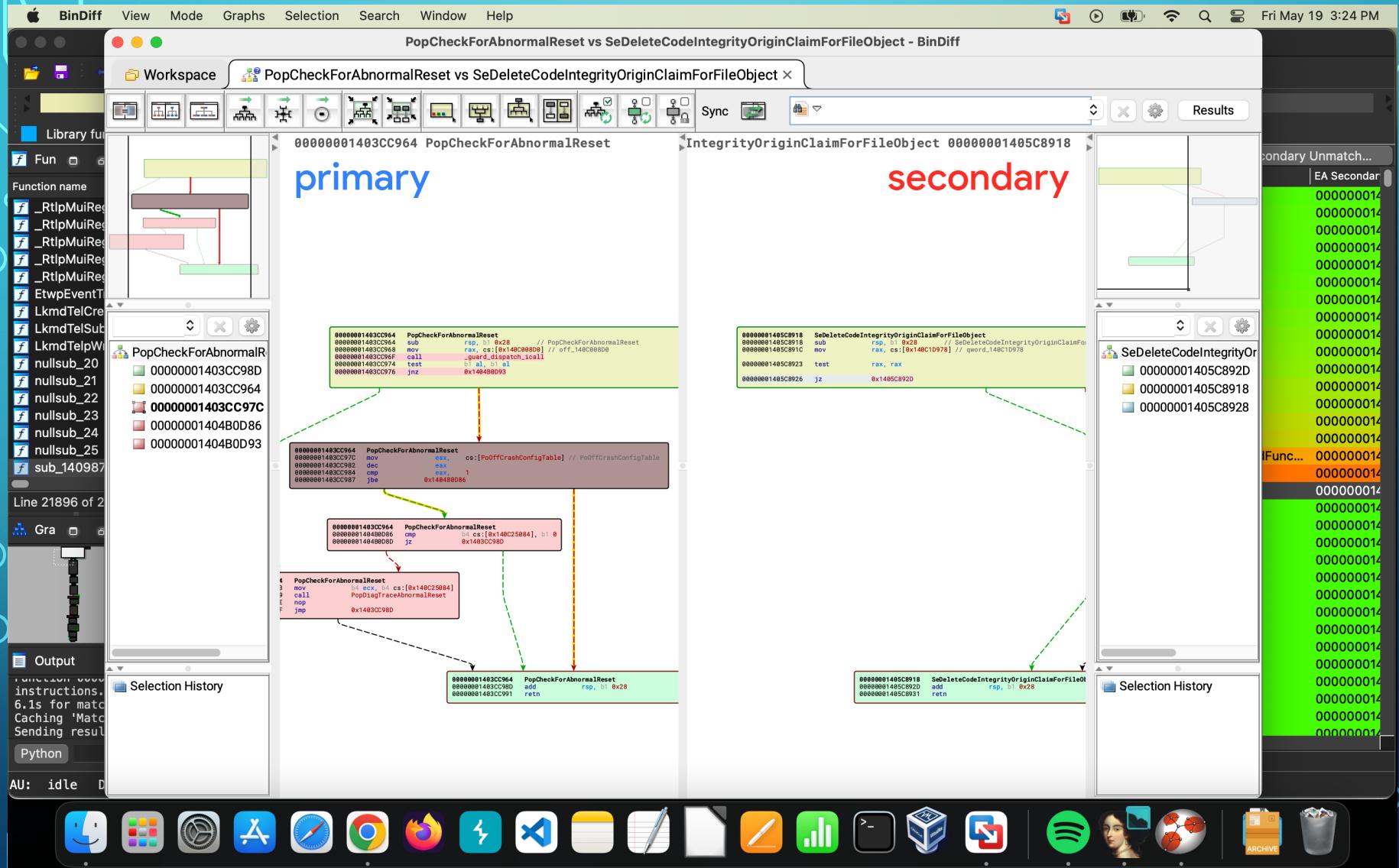
- 00000001403CC98D
- 00000001403CC964
- 00000001403CC97C
- 00000001404B0D86
- 00000001404B0D93

SeDeleteCodeIntegrityOr

- 00000001405C892D
- 00000001405C8918
- 00000001405C8928

IFunc... 00000001405C892D 00000001405C8918 00000001405C8928

AU: idle D



VULNERABLE CODE

- The patched code could be protecting a variety of attack types
 - Sanitizing input for safer usage
 - Handling data in a safer way
 - Etc
- The attack payload could be processed by the program immediately, or later on in the code

PATCH LAG

- Once you know what safeguards were added, you can use the dangerous inputs to exploit any instances of the software that have not been patched yet
 - Unfortunately, many organizations take a long time to patch, leaving those systems vulnerable to attack

PATCH LAG - IVANTI

- The vulnerable version has a particular favicon



PATCH LAG – SCREEN CONNECT

ScreenConnect 23.9.7 and below is vulnerable

SHODAN Explore Downloads Pricing ↗ "Server: ScreenConnect/23.9.7" 🔍

TOTAL RESULTS **163**

TOP COUNTRIES

COUNTRY	RESULTS
United States	115
Canada	12
United Kingdom	11
Netherlands	8
France	3
More...	

TOP PORTS

PORT	RESULTS
443	92
80	35
8040	33
8443	2
4433	1
More...	

TOP ORGANIZATIONS

Access Granted: Want to get more out of your existing Shodan account? Check out [everything you have access to](#).

ConnectWise ScreenConnect Remote Support Software ↗
73.149.191.192
c-73-149-191-192.hsd1.nh.comcast.net
Comcast IP Services, L.L.C.
United States, Portsmouth
HTTP/1.1 200 OK
Cache-Control: private
Content-Length: 22416
Content-Type: text/html; charset=utf-8
Server: ScreenConnect/23.9.7.8804-669959013 Microsoft-HTTPAPI/2.0
P3P: CP="NON CUR OUR STP STA PRE"
Date: Wed, 28 Feb 2024 10:59:57 GMT

ConnectWise ScreenConnect Remote Support Software ↗
67.233.163.25
f-67-233-163-25.dhcp.embarqhsd.net
CenturyLink Communications, LLC
United States, Ocala
HTTP/1.1 200 OK
Cache-Control: private
Content-Length: 22416
Content-Type: text/html; charset=utf-8
Server: ScreenConnect/23.9.7.8804-480294216 Microsoft-HTTPAPI/2.0
P3P: CP="NON CUR OUR STP STA PRE"
Date: Mon, 26 Feb 2024 19:38:15 GMT

OZ Planten Remote Support ↗
109.109.101.8
ip6d6d6508.static.cbizz.nl
ozplanten.nl
DELTA Fiber Nederland B.V.
Netherlands, Aalsmeer
SSL Certificate
Issued By:
- Common Name: RapidSSL TLS RSA CA G1
- Organization: DigiCert Inc
Issued To:
- Common Name: *.ozplanten.nl
Supported SSL Versions:

WINDOWS PATCHES

- Windows patches are some of the most diffed patches, because so many computers run Windows
 - Especially business computers, which usually contain very sensitive data
- However, extracting the binaries from Windows patch rollup packages can be a complicated process, involving several steps

EXTRACTING WINDOWS PATCHING

- You can obtain the Windows Patch files from the Microsoft Update Catalog

A screenshot of a web browser displaying the Microsoft Update Catalog. The URL in the address bar is catalog.update.microsoft.com/Search.aspx?q=21H1%20cumulative%202022-01%20x64. The search bar at the top contains the query "21H1 cumulative 2022-01 x64". Below the search bar, the page title is "Microsoft Update Catalog". A navigation bar includes links for "FAQ" and "help". The main content area shows a table of search results with 6 items. The first item is highlighted in yellow. The table has columns: Title, Products, Classification, Last Updated, Version, Size, and Download. The "Download" column contains blue "Download" buttons.

Title	Products	Classification	Last Updated	Version	Size	Download
2022-01 Cumulative Update Preview for Windows 10 Version 21H1 for x64-based Systems (KB5009596)	Windows 10, version 1903 and later	Updates	1/25/2022	n/a	653.3 MB	Download
2022-01 Cumulative Update Preview for .NET Framework 3.5 and 4.8 for Windows 10 Version 21H1 for x64 (KB5009467)	Windows 10, version 1903 and later	Updates	1/25/2022	n/a	65.6 MB	Download
2022-01 Cumulative Update for Windows 10 Version 21H1 for x64-based Systems (KB5010793)	Windows 10, version 1903 and later	Updates	1/17/2022	n/a	648.6 MB	Download
2022-01 Dynamic Cumulative Update for Windows 10 Version 21H1 for x64-based Systems (KB5009543)	Windows 10 GDR-DU	Security Updates	1/11/2022	n/a	635.5 MB	Download
2022-01 Cumulative Update for Windows 10 Version 21H1 for x64-based Systems (KB5009543)	Windows 10, version 1903 and later	Security Updates	1/11/2022	n/a	651.3 MB	Download
2022-01 Cumulative Update for .NET Framework 3.5 and 4.8 for Windows 10 Version 21H1 for x64 (KB5008876)	Windows 10, version 1903 and later	Security Updates	1/11/2022	n/a	65.4 MB	Download

Updates: 1 - 6 of 6 (page 1 of 1)

A screenshot of a web browser displaying the Microsoft Update Catalog. The URL in the address bar is catalog.update.microsoft.com/Search.aspx?q=21H1%20cumulative%202022-02%20x64. The search bar at the top contains the query "21H1 cumulative 2022-02 x64". Below the search bar, the page title is "Update Catalog". A navigation bar includes links for "FAQ" and "help". The main content area shows a table of search results with 5 items. The first item is highlighted in yellow. The table has columns: Title, Products, Classification, Last Updated, Version, Size, and Download. The "Download" column contains blue "Download" buttons.

Title	Products	Classification	Last Updated	Version	Size	Download
2022-02 Cumulative Update Preview for Windows 10 Version 21H1 for x64-based Systems (KB5010415)	Windows 10, version 1903 and later	Updates	2/15/2022	n/a	653.3 MB	Download
2022-02 Cumulative Update Preview for .NET Framework 3.5 and 4.8 for Windows 10 Version 21H1 for x64 (KB5010472)	Windows 10, version 1903 and later	Updates	2/15/2022	n/a	65.6 MB	Download
2022-02 Dynamic Cumulative Update for Windows 10 Version 21H1 for x64-based Systems (KB5010342)	Windows 10 GDR-DU	Security Updates	2/8/2022	n/a	640.7 MB	Download
2022-02 Cumulative Update for Windows 10 Version 21H1 for x64-based Systems (KB5010342)	Windows 10, version 1903 and later	Security Updates	2/8/2022	n/a	656.6 MB	Download
2022-02 Cumulative Update for .NET Framework 3.5 and 4.8 for Windows 10 Version 21H1 for x64 (KB5009467)	Windows 10, version 1903 and later	Updates	2/8/2022	n/a	65.6 MB	Download

Updates: 1 - 5 of 5 (page 1 of 1)

EXTRACTING WINDOWS PATCHING

- The patches are packed, and so need to be unpacked

```
PS C:\Users\student> expand.exe -F:* .\windows10.0-kb5009543-x64_c7b660efcaaa2dd1d55fe91c5cea3d9b209a15cd.msu msu  
PS C:\Users\student> expand.exe -F:* .\windows10.0-kb5010342-x64_f865479b6847db1aab8d436a37a964f31c853887.msu msu  
PS C:\Users\student> expand.exe -F:* .\Windows10.0-KB5009543-x64.cab cab1  
PS C:\Users\student> expand.exe -F:* .\Windows10.0-KB5010342-x64.cab cab2  
PS C:\Users\student> expand.exe -F:* .\Windows10.0-KB5009543-x64.cab cabA  
PS C:\Users\student> expand.exe -F:* .\Windows10.0-KB5010342-x64.cab cabB
```

EXTRACTING WINDOWS PATCHING

```
PS C:\Users\student\Desktop\patchDiffering\msu\cab1> ls

Directory: C:\Users\student\Desktop\patchDiffering\msu\cab1

Mode                LastWriteTime        Length  Name
----                -----        ----  -
d-----      5/19/2023    9:50 AM           0  cabA
-----      1/8/2022     3:27 AM      15318363  SSU-19041.1371-x64.cab
-----      1/8/2022     3:28 AM          202  toc.xml
-----      1/8/2022     3:28 AM          8819  update.cat
-----      1/8/2022     3:28 AM         421548  update.mum
-----      1/8/2022     3:27 AM      650697678  Windows10.0-KB5009543-x64.cab
```

```
PS C:\Users\student\Desktop\patchDiffering\msu\cab1\cabA> ls

Directory: C:\Users\student\Desktop\patchDiffering\msu\cab1\cabA

Mode                LastWriteTime        Length  Name
----                -----        ----  -
d-----      5/19/2023   10:08 AM           0  update
-----      1/7/2022    10:12 PM          93  cabinet.cablist.ini
-----      1/7/2022    10:12 PM      98808009  Cab_1_for_KB5009543_PSFX.cab
-----      1/7/2022    10:08 PM      563004122  Cab_2_for_KB5009543_PSFX.cab
-----      1/7/2022    9:25 PM          8819  update.cat
-----      1/7/2022    9:23 PM         421548  update.mum
```

EXTRACTING WINDOWS PATCHING

- Since Windows patches are cumulative (they contain all of the fixes from previous patches as well), we need to extract the delta (change) between the latest and second latest patches
 - We will be using a delta extraction script written by Jaime Geiger, a course author of the Sans 760 Advanced Exploit Development for Penetration Testers course

EXTRACTING WINDOWS PATCHING

```
PS C:\Users\student\Desktop\patchDiffing\msu\cab1\cabA> gci -rec C:\Windows\WinSxS\ -Filter ntoskrnl.exe

Directory:
C:\Windows\WinSxS\amd64_microsoft-windows-os-kernel_31bf3856ad364e35_10.0.19041.1566_none_e0d5c562696cbc64

Mode           LastWriteTime       Length Name
----           -----          -----
-a---  2/17/2022 11:43 AM      10849616 ntoskrnl.exe

Directory:
C:\Windows\WinSxS\amd64_microsoft-windows-os-kernel_31bf3856ad364e35_10.0.19041.1566_none_e0d5c562696cbc64\f

Mode           LastWriteTime       Length Name
----           -----          -----
-a---  2/12/2022 1:44 AM       1574157 ntoskrnl.exe

Directory:
C:\Windows\WinSxS\amd64_microsoft-windows-os-kernel_31bf3856ad364e35_10.0.19041.1566_none_e0d5c562696cbc64\r

Mode           LastWriteTime       Length Name
----           -----          -----
-a---  2/12/2022 1:44 AM       1340897 ntoskrnl.exe
```

```
PS C:\Users\student> python C:\lab\day4\delta_patch.py -i ntoskrnl.exe -o ntoskrnl_2022-01.exe C:\Users\student\Desktop\patchDiffing\r\ntoskrnl.exe C:\Users\student\Desktop\patchDiffing\msu\cab1\cabA\update\amd64_microsoft-windows-os-kernel_31bf3856ad364e35_10.0.19041.1466_none_e0e0954e6964a073\f\ntoskrnl.exe
```

```
PS C:\Users\student> python C:\lab\day4\delta_patch.py -i ntoskrnl.exe -o ntoskrnl_2022-02.exe C:\Users\student\Desktop\patchDiffing\r\ntoskrnl.exe C:\Users\student\Desktop\patchDiffing\msu\cab2\cabB\update\amd64_microsoft-windows-os-kernel_31bf3856ad364e35_10.0.19041.1526_none_e0d1c43a69705708\f\ntoskrnl.exe
```

EXPLOIT DEVELOPMENT

- Note that not all vulnerabilities are the same level of easiness/difficulty to exploit
 - Some are easy, like a filter that misses removing a special character
 - just send an input with that special character
 - Others are very difficult to exploit
 - a memory region that needs to be groomed via multiple inputs, in order to get the needed values into memory at the needed place

POC

- I wrote a simple program with a buffer overflow
 - Since trying to exploit a patched Windows vulnerability can be too long and complex for a presentation
- Normally, the program runs the “ls” command, which prints the directory

```
deadlist@sec760:~/poc$ ./bufferOverflow AAAAAAAA
bufferOverflow  bufferOverflow.c  bufferOverflowFixed  bufferOverflow_fixed.c
deadlist@sec760:~/poc$
```

POC

- But when your input overflows into the “ls” string memory, you can replace “ls” with anything you want
 - Print the /etc/shadow file

```
deadlist@sec760:~/poc$ ./bufferOverflow AAAAAAAAAA"sudo cat /etc/shadow"
root:$6$Pw/srH4fD7dsHnRm$5.yQsUvPLSqGWMdv4S9G/o2j3Z2midj582/gH148CLNiZsRu9py3hgj
/XyvRQVGZJsegkcyrIRlFxegiZTLVs1:19454:0:99999:7:::
daemon:*:18002:0:99999:7:::
bin:*:18002:0:99999:7:::
sys:*:18002:0:99999:7:::
sync:*:18002:0:99999:7:::
games:*:18002:0:99999:7:::
man:*:18002:0:99999:7:::
lp:*:18002:0:99999:7:::
mail:*:18002:0:99999:7:::
```

POC

```
//Poc c program that is vulnerable to a buffer overflow from the buffer variable into the
//modified variable; and then executes the contents of the modified variable.
//The buffer overflow allows you to change the modified variable from "ls" to another
//command.

//*note: to run a command that has spaces, you need to enclose it in quotes
//AAAAAAA"cat /etc/pass"
#include <stdlib.h>
#include <unistd.h>
#include <stdio.h>
#include <string.h>

int main(int argc, char **argv)
{
    char modified[100] = "";
    strcat(modified, "ls");
    char buffer[10];

    if(argc == 1) {
        printf(1, "please specify an argument\n");
    }

    strcpy(buffer, argv[1]);
    system(modified);
}
```

POC

- BinDiff shows the strcpy function is only used in the vulnerable version of the program
 - The strcpy function doesn't do input length checking, allowing the buffer overflow into the “ls” memory

The screenshot shows the IDA Pro interface with the following details:

- Functions:** Shows `_init_proc` and `sub_1020`.
- Hex View-A:** Displays assembly code for the `main` function:

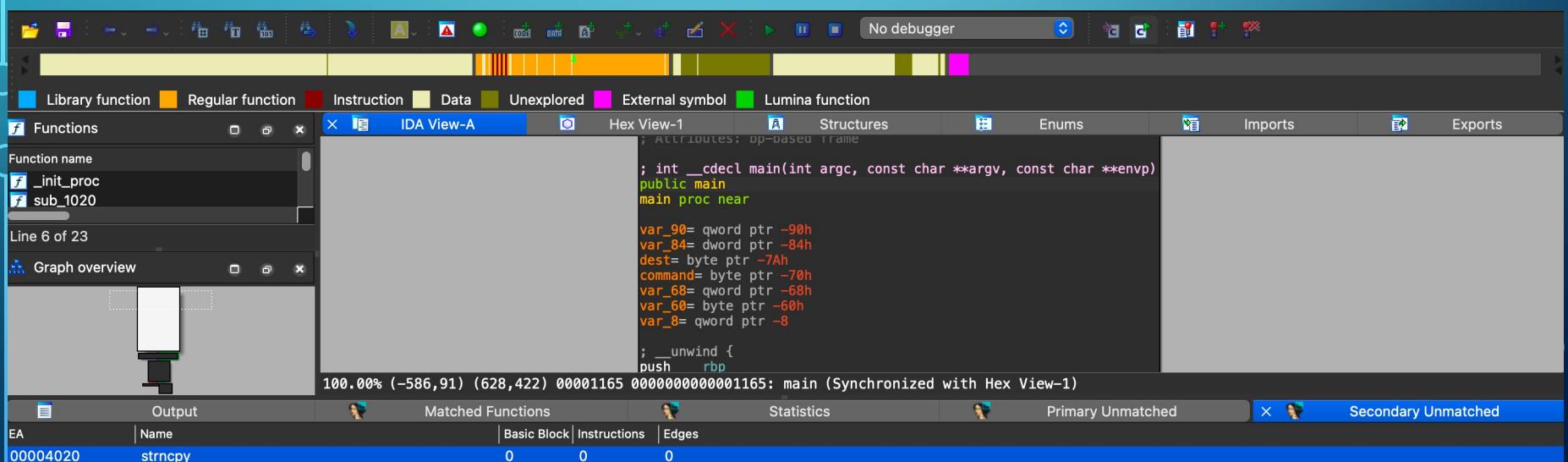
```
; int __cdecl main(int argc, const char **argv, const char **envp)
public main
main proc near

var_90= qword ptr -90h
var_84= dword ptr -84h
dest= byte ptr -7Ah
command= byte ptr -70h
var_68= qword ptr -68h
var_60= byte ptr -60h
var_8= qword ptr -8

; __unwind {
push    rbp
```
- Graph overview:** A graph showing the control flow of the program.
- Output:** Shows the address `00004020` and the name `strcpy`.
- Matched Functions:** Basic Block, Instructions, Edges.
- Statistics:** Primary Unmatched (selected tab), Secondary Unmatched.

POC

- However, BinDiff shows the `strncpy` function is used in the patched version of the program
 - The `strncpy` function does do input length checking
 - This prevents long user input from overflowing into the "ls" memory



The screenshot shows the BinDiff application interface. The top menu bar includes File, Edit, View, Tools, Options, Help, and a status message "No debugger". Below the menu is a toolbar with various icons. The main window has tabs for Functions, IDA View-A (selected), Hex View-1, Structures, Enums, Imports, and Exports. The Functions tab lists `_init_proc` and `sub_1020`. The IDA View-A tab displays assembly code for the `main` function:

```
; int __cdecl main(int argc, const char **argv, const char **envp)
public main
main proc near

var_90= qword ptr -90h
var_84= dword ptr -84h
dest= byte ptr -7Ah
command= byte ptr -70h
var_68= qword ptr -68h
var_60= byte ptr -60h
var_8= qword ptr -8

; _unwind {
push rbp
```

The status bar at the bottom shows "100.00% (-586,91) (628,422) 00001165 0000000000001165: main (Synchronized with Hex View-1)". The bottom navigation bar includes Output, Matched Functions, Statistics, Primary Unmatched, Secondary Unmatched, and tabs for EA, Name, Basic Block, Instructions, and Edges. A table at the bottom tracks the `strncpy` function with values 0, 0, 0.

POC

```
//Poc c program that is vulnerable to a buffer overflow from the buffer variable into the
//modified variable; and then executes the contents of the modified variable.
//The buffer overflow allows you to change the modified variable from "ls" to another
//command.
//*note: to run a command that has spaces, you need to enclose it in quotes
//AAAAAAA"cat /etc/pass"
#include <stdlib.h>
#include <unistd.h>
#include <stdio.h>
#include <string.h>

int main(int argc, char **argv)
{
    char modified[100] = "";
    strcat(modified, "ls");
    char buffer[10];

    if(argc == 1) {
        printf(1, "please specify an argument\n");
    }

    strncpy(buffer, argv[1], 10);
    system(modified);
```

POC

- The fixed version of the program doesn't print the whoami command results when you overflow the buffer

```
lab@lab:~/Desktop/patchDiff_poc$ ./bufferOverflow AAAAAAAAAAwhoami  
lab
```

```
lab@lab:~/Desktop/patchDiff_poc$ ./bufferOverflow_fixed AAAAAAAA  
bufferOverflow bufferOverflow.c bufferOverflow_fixed bufferOverflow_fixed.c  
lab@lab:~/Desktop/patchDiff_poc$ ./bufferOverflow_fixed AAAAAAAAAAwhoami  
bufferOverflow bufferOverflow.c bufferOverflow_fixed bufferOverflow_fixed.c
```

TEASER FOR NEXT YEAR – ZERO DAY BUG HUNTING

- Look for dangerous functions, that don't do bounds checking when handling data
 - `strcpy()`, `strcat()`, `sprintf()`, `gets()`, etc
 - Then walk up the code chain to see if the dangerous function receives user inputted data
- Inspired by a talk that Stephen Sims gave at Sans Hackfest a couple of years ago on looking for zero days
 - I was like “Wow, it can be that easy?”

ANY QUESTIONS?



dreamstime.com

ID 107284360 © Dreammasterphotographer